e-ISSN:2582-5208
International Research Journal of Modernization in Engineering Technology and Science
Volume:03/Issue:06/June-2021 Impact Factor- 5.354 www.irjmets.com

HYBRID INDEXED AND LINKED LIST FILE ALLOCATION SYSTEM

Shaik Haseeb Ur Rahman[*1]**, Ayesha Siddiqua**[*2]

[*1]Student, Department of Computer Science and Engineering, Vellore Institute of Technology, Vellore, Tamil Nadu, India-632014.

[*2]Student, Department of Mechanical Engineering, Vellore Institute of Technology, Vellore, Tamil Nadu, India-632014.

ABSTRACT

Without a proper management, the information in the storage medium would be a large body of data with no way to tell where an information starts or ends. By taking the data or information separating and organizing them the information can be easily isolated and identified and this is done through file systems using the system calls. Each group of data is called as file and the collection of algorithms and data structures which performs the translation from logical file operations (system calls) to actual physical storage of information is known as File System. The ways through which the files are stored are known as File allocation. The File Allocation Table of file systems is a way in which the data is stored in a table format and in which some disadvantages of normal file allocation methods will be overcome. In this paper, we are trying to improve the drawbacks of indexed and linked list file allocation system by proposing a new file allocation system.

Keywords: File Allocation, Disk Space Management, Paging, Memory Management, File Allocation Table.

I. INTRODUCTION

In this paper, our team is trying to implement the various methods of file systems through file allocation and to improve the drawbacks of indexed and linked list file allocation system by proposing a new file allocation system in operating systems. The collection of algorithms and data structures which performs the translation from logical file operations (system calls) to actual physical storage of information is known as File System. The File System creates, manipulates, renames, copies and remove files to and from a storage device. It also organizes files into common storage units, keeps track of all files and directory locations, assists users and relate files and folders to the physical structure of the storage medium. File system is implemented through the following:

- **Contiguous Allocation**
- **Linked list Allocation**
- **Indexed Block Allocation**
- **File Allocation Table**

II. METHODOLOGY

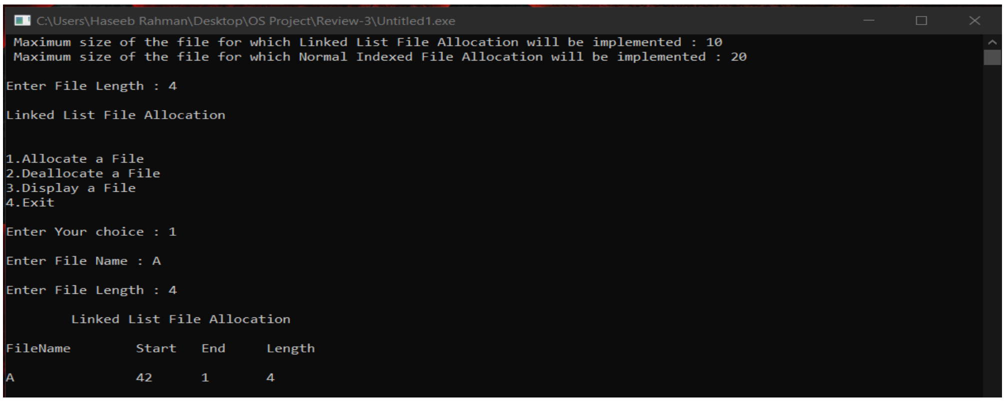
Comparing all the three-file allocation we conclude that the best file allocation method is Indexed file allocation. The Contiguous File Allocation method suffers a lot of external fragmentation but it's access time is and lesser than Linked list File allocation method as it's uses the concept of direct access of files whereas Linked List File Allocation method doesn't suffers from external fragmentation but it has a larger file access time as it follows sequential access method. Indexed File Allocation Method is free of external fragmentation and it uses the random-access method to access a file which is faster in all three allocation methods. Hence, the Indexed file allocation is comparatively the best file allocation method in all the three file allocation methods. But the main problems in the Indexed file allocation method is that for a large size file, it is very difficult for single index block to hold all the pointers. Hence, we can't store large files using this method and For very small files say files that expend only 2-3 blocks the indexed allocation would keep on the entire block for the pointers which is insufficient in terms of memory utilization and hence it suffers from a wastage of space. To minimize the disadvantages of all the three-file allocation method we need a new improvised or hybridized file allocation method and the solution to problem given statement above is given below.

III. MODELING AND ANALYSIS

As we already saw that the Indexed File Allocation Method is the best file allocation method in all three File Allocation methods and also that the Linked List File Allocation method is better than Contiguous File Allocation method, So it will be a good idea if we try to combine the Linked List File Allocation method and Indexed File Allocation Method. Our File allocation method will basically implement Linked List File Allocation method, but we will fix a number for the size of the file after which Indexed File Allocation Method implementation takes place. The main advantage of this method will be that very small files say files that expend only 2-3 blocks the indexed allocation would not keep on the entire block for the pointers which is insufficient in terms of memory utilization and hence it will not suffer from wastage of space. Our next idea for our proposed File Allocation method is that instead of having one indexed block we will have two indexed blocks in which half of the pointers of the allocated blocks will be stored in one indexed block while the other half pointers of the allocated blocks will be stored in another indexed block. The advantage of this method will be that the problem of not being able to store a large file due to not being capable to store all the pointers in one indexed block will be terminated.

IV. RESULTS AND DISCUSSION

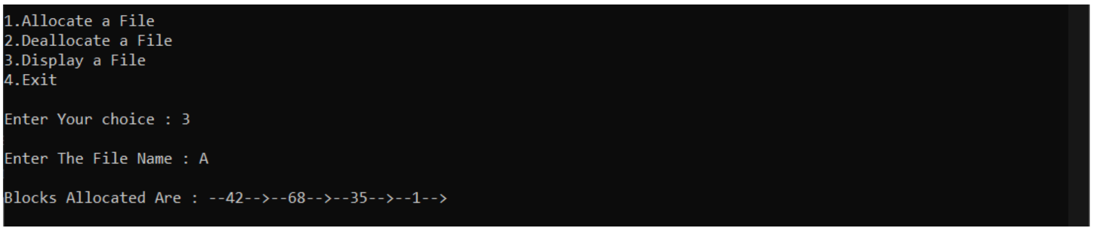
We discussed earlier that the main criteria to be look upon for an efficient file allocation method is low external fragmentation and low access time of the file and also that the index allocation is better among all the three file allocation methods. So, here we tried to reduce the disadvantages of the indexed file allocation method. The main disadvantages of Indexed File Allocation method are that for small files allocation the memory gets wasted in the indexed block and for large files the indexed block cannot occupy all the pointers of the blocks. Hence, we tried to remove these disadvantages by implementing linked list file allocation method for small files, normal indexed file allocation for medium sized files and modified indexed file allocation method for large files. We will ask the user for maximum size of the block up to which Linked list file allocation and maximum file size up to which indexed file allocation. Suppose, the user enters 10 as the maximum size of the block up to which Linked list file allocation will be implemented and 20 as the maximum size of the block up to which Indexed file allocation will be implemented and hence after 20 size of our file our modified file allocation method will be implemented. Hence, for the file size less than 10 normal linked list file allocation will be implemented, for file size in between 10 and 20 normal indexed file allocation and for file size greater than 20 our proposed modified file allocation method will be implemented which has two indexed blocks for storing larger files all pointers. Thus, this is how we minimized the external fragmentation and the file access time in our new proposed file allocation method and made our file allocation more efficient.



C:\Users\Haseeb Rahman\Desktop\OS Project\Review-3\Untitled1.exe

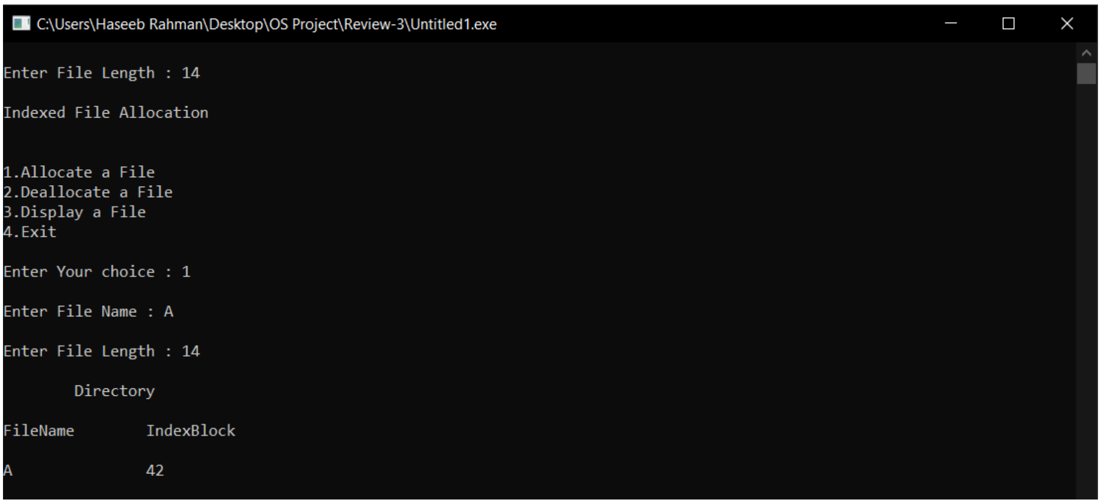
```
 Maximum size of the file for which Linked List File Allocation will be implemented : 10
 Maximum size of the file for which Normal Indexed File Allocation will be implemented : 20

Enter File Length : 4

Linked List File Allocation


1.Allocate a File
2.Deallocate a File
3.Display a File
4.Exit

Enter Your choice : 1

Enter File Name : A

Enter File Length : 4

        Linked List File Allocation

FileName        Start   End     Length

A               42      1       4
```

Figure 1: File size lower than the lower threshold size



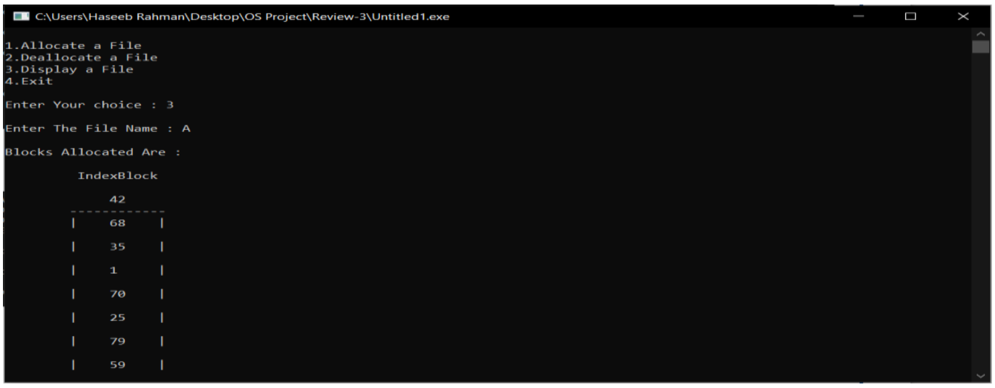
```
1.Allocate a File
2.Deallocate a File
3.Display a File
4.Exit

Enter Your choice : 3

Enter The File Name : A

Blocks Allocated Are : --42-->--68-->--35-->--1-->
```

Figure 2: File allocated of size lower than the lower threshold size


C:\Users\Haseeb Rahman\Desktop\OS Project\Review-3\Untitled1.exe

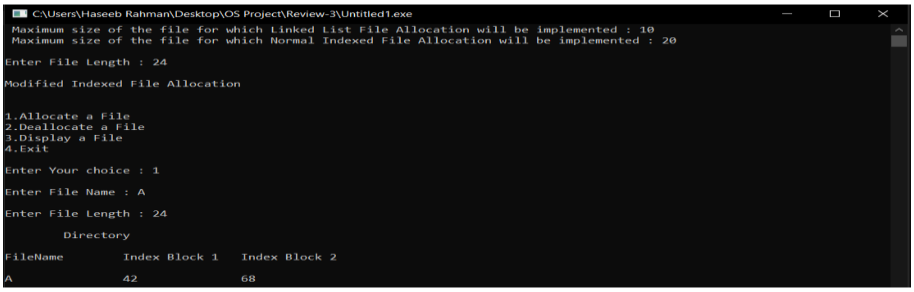
```
Enter File Length : 14

Indexed File Allocation


1.Allocate a File
2.Deallocate a File
3.Display a File
4.Exit

Enter Your choice : 1

Enter File Name : A

Enter File Length : 14

        Directory

FileName        IndexBlock

A               42
```

Figure 3: File size greater than the lower threshold size but higher than greater threshold size


C:\Users\Haseeb Rahman\Desktop\OS Project\Review-3\Untitled1.exe

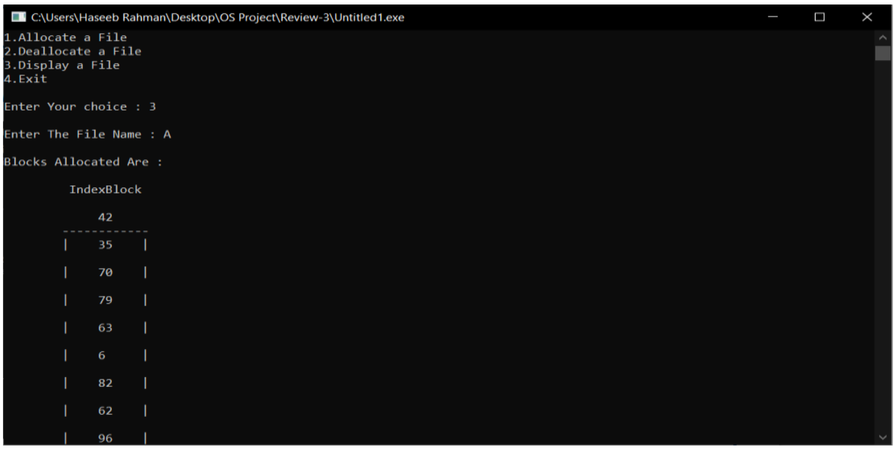
```
1.Allocate a File
2.Deallocate a File
3.Display a File
4.Exit

Enter Your choice : 3

Enter The File Name : A

Blocks Allocated Are :

         IndexBlock

            42
        ------------
        |   68     |

        |   35     |

        |   1      |

        |   70     |

        |   25     |

        |   79     |

        |   59     |
```

Figure 4: File allocated of size greater than the lower threshold size but higher than greater threshold size

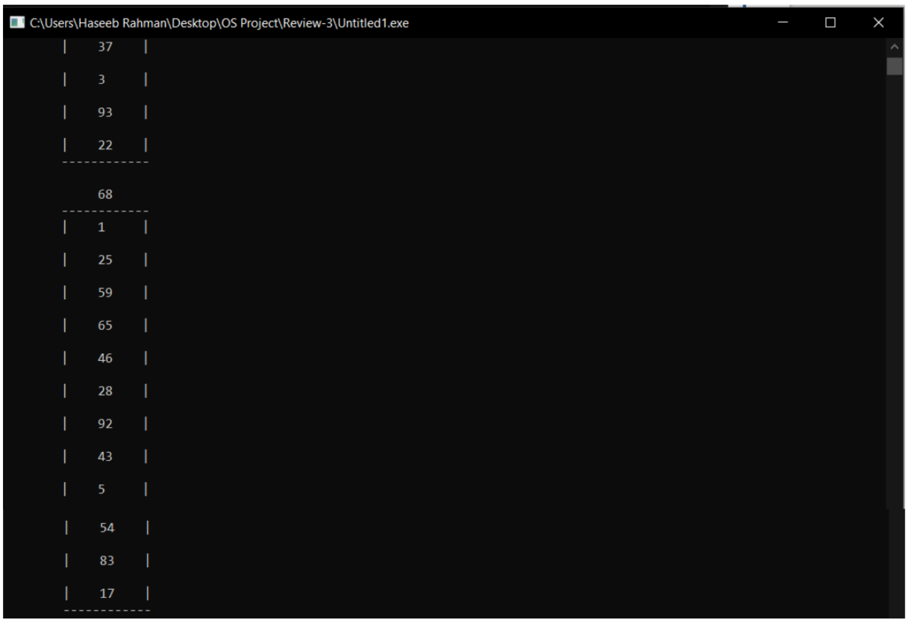

C:\Users\Haseeb Rahman\Desktop\OS Project\Review-3\Untitled1.exe

```
 Maximum size of the file for which Linked List File Allocation will be implemented : 10
 Maximum size of the file for which Normal Indexed File Allocation will be implemented : 20

Enter File Length : 24

Modified Indexed File Allocation


1.Allocate a File
2.Deallocate a File
3.Display a File
4.Exit

Enter Your choice : 1

Enter File Name : A

Enter File Length : 24

        Directory

FileName        Index Block 1   Index Block 2

A               42              68
```

Figure 5: File size greater than higher than greater threshold size


C:\Users\Haseeb Rahman\Desktop\OS Project\Review-3\Untitled1.exe

```
1.Allocate a File
2.Deallocate a File
3.Display a File
4.Exit

Enter Your choice : 3

Enter The File Name : A

Blocks Allocated Are :

         IndexBlock

            42
        -----------
        |   35    |

        |   70    |

        |   79    |

        |   63    |

        |   6     |

        |   82    |

        |   62    |

        |   96    |
```



```
C:\Users\Haseeb Rahman\Desktop\OS Project\Review-3\Untitled1.exe
   |   37   |

   |   3    |

   |   93   |

   |   22   |
   ----------

       68
   ----------
   |   1    |

   |   25   |

   |   59   |

   |   65   |

   |   46   |

   |   28   |

   |   92   |

   |   43   |

   |   5    |

   |   54   |

   |   83   |

   |   17   |
   ----------
```

Figure 6: File allocated of size greater than higher than greater threshold size

V. CONCLUSION

The main criteria to be look upon for a file allocation method is fragmentation and access time of the file. So, with the reference of the defined file allocation methods we will tried to make a new improvised file allocation method which mainly concentrates on reducing the fragmentation and the access time of file. This can be done by making a hybrid of two allocation methods and improvising some elements of the hybridized file allocation method. Hence, after considerate optimizations we were able to achieve to propose a new hybrid file allocation method which eliminates the drawbacks of the above-mentioned algorithms.

VI. REFERENCES

[1] https://www.slideshare.net/shashikantpabari/file-system-and-file-allocationtablesystem

[2] https://www.geeksforgeeks.org/file-allocation-methods/

[3] https://program24x7.blogspot.com/2017/08/linked-file-allocation.html

[4] https://whatis.techtarget.com/definition/file-allocation-table-FAT

[5] www.google.comsearchq=fat+file+system&source=lnms&tbm=isch&sa=X&ved=0ahUKwi
_tm_8KjkAhWnIbcAHVgABoYQ_AUIESgB&biw=738&bih=432.